

## CLAIMS

What is claimed is:

- 1           1.     A method to efficiently design and implement a matched  
2 instruction set processor system, including:  
3           decomposing the matched instruction set processor system into  
4 interconnected design vectors; and  
5           analyzing and mapping the interconnected design vectors into specific  
6 hardware and software elements.
- 1           2.     The method of claim 1, wherein decomposing the matched  
2 instruction set processor into interconnected design vectors includes:  
3           performing a concurrency analysis to determine concurrency of execution  
4 of actors.
- 1           3.     The method of claim 2, wherein performing a concurrency analysis  
2 includes:  
3           analyzing an order of invocation of the actors to determine an invocation  
4 type, wherein the invocation type is one of coterminous invocation, sequential  
5 invocation, and overlapping pipelined invocation.
- 1           4.     The method of claim 1, wherein decomposing the matched  
2 instruction set processor into interconnected design vectors includes:  
3           performing a concurrency analysis to identify an invocation period, the  
4 invocation period representing a time duration between two successive  
5 invocations.
- 1           5.     The method of claim 1, wherein decomposing the matched  
2 instruction set processor into interconnected design vectors includes:

3 performing a concurrency analysis to identify a maximum allowable  
4 response time, the maximum allowable response time representing a maximum  
5 time duration during which processing should be completed.

1 6. The method of claim 1, wherein decomposing the matched  
2 instruction set processor system into interconnected design vectors includes:  
3 performing model atomization to convert an actor into application specific  
4 code.

1 7. The method of claim 6, wherein performing model atomization to  
2 convert an actor into application specific code includes:  
3 removing parameters by converting the parameters to private data  
4 constants inside the code; and  
5 removing data polymorphism and domain polymorphism.

1 8. The method of claim 1, wherein decomposing the matched  
2 instruction set processor system into interconnected design vectors includes:  
3 performing functional vector creation to put an actor into design vector  
4 format.

1 9. The method of claim 8, wherein performing functional vector  
2 creation to put an actor into design vector format includes:  
3 creating a design vector;  
4 putting a fire() method into the design vector;  
5 creating a header data of the design vector;  
6 generating the trailer data of the design vector; and  
7 creating binding methods for the design vector.

1           10.    The method of claim 1, wherein decomposing the matched  
2 instruction set processor system into interconnected design vectors includes:  
3           performing a model test and verification.

1           11.    The method of claim 1, wherein decomposing the matched  
2 instruction set processor system into interconnected design vectors includes:  
3           performing a functional vector extraction to extract source code for  
4 functional vectors from encapsulated actors.

1           12.    The method of claim 1, wherein decomposing the matched  
2 instruction set processor system into interconnected design vectors includes:  
3           performing an interconnect vector extraction to extract source code for  
4 interconnect vectors from encapsulated actors.

1           13.    The method of claim 1, wherein decomposing the matched  
2 instruction set processor system into interconnected design vectors includes:  
3           performing a Conjugate Virtual Machine (CVM) generation to extract  
4 CVM instructions.

1           14.    The method of claim 1, wherein decomposing the matched  
2 instruction set processor system into interconnected design vectors includes:  
3           performing a stand-alone testing of extracted design vectors.

1           15.    A machine-readable medium comprising instructions which, when  
2 executed by a machine, cause the machine to perform operations comprising:  
3           decomposing the matched instruction set processor system into  
4 interconnected design vectors; and  
5           analyzing and mapping the interconnected design vectors into specific  
6 hardware and software elements.

1           16.    The machine-readable medium of claim 15, wherein decomposing  
2   the matched instruction set processor into interconnected design vectors  
3   includes:  
4           performing a concurrency analysis to determine an execution order of  
5   actors.

1           17.    The machine-readable medium of claim 15, wherein decomposing  
2   the matched instruction set processor system into interconnected design vectors  
3   includes:  
4           performing model atomization to convert an actor into application specific  
5   code.

1           18.    The machine-readable medium of claim 15, wherein decomposing  
2   the matched instruction set processor system into interconnected design vectors  
3   includes:  
4           performing functional vector creation to put an actor into design vector  
5   format.

1           19.    The machine-readable medium of claim 15, wherein decomposing  
2   the matched instruction set processor system into interconnected design vectors  
3   includes:  
4           performing a model test and verification.

1           20.    The machine-readable medium of claim 15, wherein decomposing  
2   the matched instruction set processor system into interconnected design vectors  
3   includes:  
4           performing a functional vector extraction to extract source code for  
5   functional vectors from encapsulated actors.

1           21.    The machine-readable medium of claim 15, wherein decomposing  
2   the matched instruction set processor system into interconnected design vectors  
3   includes:

4           performing an interconnect vector extraction to extract source code for  
5   interconnect vectors from encapsulated actors.

1           22.    The machine-readable medium of claim 15, wherein decomposing  
2   the matched instruction set processor system into interconnected design vectors  
3   includes:

4           performing a Conjugate Virtual Machine (CVM) generation to extract  
5   CVM instructions.

1           23.    The machine-readable medium of claim 15, wherein decomposing  
2   the matched instruction set processor system into interconnected design vectors  
3   includes:

4           performing a stand-alone testing of extracted design vectors.

1           24.    An architectural modeling apparatus to decomposing a matched  
2   instruction set processor system into interconnected design vectors, comprising:

3           a concurrency analyzer to determine an execution order of actors; and  
4           a model atomizer to convert the actors into application specific code.

1           25.    The architectural modeling apparatus of claim 24, further  
2   comprises:

3           a functional vector creator to put the actors into design vector format.

1           26.    The architectural modeling apparatus of claim 24, further  
2   comprises:

3           a testing unit to verify a model.

1           27.    The architectural modeling apparatus of claim 24, further  
2 comprises:  
3           a functional vector extractor to extract source code for functional vectors  
4 from encapsulated actors.

1           28.    The architectural modeling apparatus of claim 24, further  
2 comprises:  
3           an interconnect vector extractor to extract source code for interconnect  
4 vectors from encapsulated actors.

1           29.    The architectural modeling apparatus of claim 24, further  
2 comprises:  
3           a Conjugate Virtual Machine (CVM) generator to extract CVM  
4 instructions.